

Art Unit: 2192

**DETAILED ACTION**

1. This Office action is responsive to applicant's amendment and response received January 20, 2010.
2. Claim 1, 7, 10, 18, 21, and 23 have been amended, and claims 6, 17, and 22 have been cancelled by Applicant. Claims 1, 8, 10, 19, 21, and 24 have been amended by Examiner's Amendment. Claims 7, 18, and 23 have been cancelled by Examiner's Amendment. Claims 1-5, 8-10, 12-16, 19-21, 24 and 25 are pending and are addressed in this office action.

***Response to Amendment***

3. The amendments to the claims and applicant's arguments overcome the objection and rejections set forth in the previous Action. The objections and rejections have been withdrawn.

***Examiner's Amendment***

4. An examiner's amendment to the record appears below. Should the changes and/or additions be unacceptable to applicant, an amendment may be filed as provided by 37 CFR 1.312. To ensure consideration of such an amendment, it MUST be submitted no later than the payment of the issue fee.
5. Authorization for this examiner's amendment was given in a telephone interview with Peter B. Martine on April 22, 2010.
6. The claims are amended, as presented below, to adopt the changes that Examiners provided to Applicant's representative on April 20, 2010 (see attached facsimile transmission). Claim 21 has been further amended to adopt changes discussed during the telephone interview with regard to directing the claim to statutory subject matter. The amendments to the claims distinguish the claims over the prior art of record.

Art Unit: 2192

**IN THE CLAIMS:**

Please cancel claims 7, 18, and 23, and amendment claims 1, 8, 10, 19, 21, and 24.

This listing of claims will replace all prior versions and listings of claims in the application:

**Listing of claims:**

**Claim 1 (Currently Amended):** A method for the controlled execution of a program, the program being intended for a virtual machine, on a portable data carrier, wherein

- the data carrier has a processor which executes at least a first and a second virtual machine for each execution of the program,
- the program is executed both by the first and by the second virtual machine,
- the first and the second virtual machine both access a common heap in a non-volatile memory of the data carrier, ~~wherein write operations to the common heap are only performed by one of the first and second virtual machines,~~ wherein, when an instruction of the program that contains a write operation to the common heap is being executed, a write operation is performed only by the first virtual machine.
- an operating state of the first virtual machine and an operating state of the second virtual machine are checked during execution of the program for correspondence, and
- execution of the program is aborted if a difference is found between the operating state of the first virtual machine and the operating state of the second virtual machine.

**Claim 2 (Previously Presented):** A method according to claim 1, wherein checking of the operating state of the first virtual machine and of the operating state of the second virtual machine for correspondence comprises checking whether the state of a program counter of the first virtual machine is the same as the state of a program counter of the second virtual machine.

Art Unit: 2192

**Claim 3 (Previously Presented):** A method according to claim 1, wherein checking of the operating state of the first virtual machine and of the operating state of the second virtual machine for correspondence comprises checking whether the level of a stack pointer of the first virtual machine is the same as the level of a stack pointer of the second virtual machine.

**Claim 4 (Previously Presented):** A method according to claim 1, wherein checking of the operating state of the first virtual machine and the operating state of the second virtual machine for correspondence comprises checking whether a value of the most recent element in a stack associated with the first virtual machine is the same as a value of the most recent element in a stack associated with the second virtual machine.

**Claim 5 (Previously Presented):** A method according to claim 1, wherein checking of the operating state of the first virtual machine and of the operating state of the second virtual machine for correspondence is in each case performed after an instruction of the program has been executed both by the first and by the second virtual machine.

**Claim 6 (Canceled).**

**Claim 7 (Canceled).**

**Claim 8 (Currently Amended):** A method according to claim 17, wherein the instruction of the program is executed first by the first virtual machine and then by the second virtual machine, and, instead of performing the write operation, the second virtual machine checks whether a value that is to be written is present in the heap at the location that is to be written to.

**Claim 9 (Previously Presented):** A method according to claim 1, wherein the program is a *Java Card Applet* intended for execution by a JCVM (*Java Card Virtual Machine*).

**Claim 10 (Currently Amended):** A portable data carrier, having a processor, a non-volatile memory, an operating system, at least a first and a second virtual machine, and a program, wherein

- the processor executes both the first and second virtual machine,
- the program is executed both by the first and by the second virtual machine for each execution of the program,
- the first and the second virtual machine both access a common heap in the non-volatile memory of the data carrier, ~~wherein write operations to the common heap are only performed by one of the first and second virtual machines,~~ wherein, when an instruction of the program that contains a write operation to the common heap is being executed, the write operation is performed only by the first virtual machine,
- the operating system controls the processor to check the operating state of the first virtual machine and the operating state of the second virtual machine during execution of the program for correspondence, and
- the operating system controls the processor to abort execution of the program if a difference is found between the operating state of the first virtual machine and the operating state of the second virtual machine.

**Claim 11 (Canceled).**

**Claim 12 (Previously Presented):** A portable data carrier according to claim 10, wherein the data carrier is one of a chip card and a chip module.

Art Unit: 2192

**Claim 13 (Previously Presented):** A portable data carrier according to claim 10, wherein checking of the operating state of the first virtual machine and of the operating state of the second virtual machine for correspondence comprises checking whether the state of a program counter of the first virtual machine is the same as the state of a program counter of the second virtual machine.

**Claim 14 (Previously Presented):** A portable data carrier according to claim 10, wherein checking of the operating state of the first virtual machine and of the operating state of the second virtual machine for correspondence comprises checking whether the level of a stack pointer of the first virtual machine is the same as the level of a stack pointer of the second virtual machine.

**Claim 15 (Previously Presented):** A portable data carrier according to claim 10, wherein checking of the operating state of the first virtual machine and the operating state of the second virtual machine for correspondence comprises checking whether the value of the most recent element in a stack associated with the first virtual machine is the same as the value of the most recent element in a stack associated with the second virtual machine.

**Claim 16 (Previously Presented):** A portable data carrier according to claim 10, wherein checking of the operating state of the first virtual machine and of the operating state of the second virtual machine for correspondence is in each case performed after an instruction of the program has been executed both by the first and by the second virtual machine.

**Claim 17 (Canceled).**

**Claim 18 (Canceled).**

Art Unit: 2192

**Claim 19 (Currently Amended):** A portable data carrier according to claim 1048, wherein the instruction of the program is executed first by the first virtual machine and then by the second virtual machine, and, instead of performing the write operation, the second virtual machine checks whether the value that is to be written is present in the heap at the location that is to be written to.

**Claim 20 (Previously Presented):** A portable data carrier according to claim 10, wherein the program is a *Java Card Applet* intended for execution by a JCVM (*Java Card Virtual Machine*).

**Claim 21 (Currently Amended):** A computer storage program product having program instructions for causing a processor of a portable data carrier to perform a method for the controlled execution of a program, the program being intended for a virtual machine, wherein

- the processor executes at least a first and a second virtual machine,
- the program is executed both by the first and by the second virtual machine for each execution of the program,
- the first and the second virtual machine both access a common heap in a non-volatile memory of the data carrier, ~~wherein write operations to the common heap are only performed by one of the first and second virtual machines,~~ wherein, when an instruction of the program that contains a write operation to the common heap is being executed, the write operation is performed only by the first virtual machine,

- the operating state of the first virtual machine and the operating state of the second virtual machine are checked during execution of the program for correspondence, and
- execution of the program is aborted if a difference is found between the operating state of the first virtual machine and the operating state of the second virtual machine.

**Claim 22 (Canceled).**

Art Unit: 2192

**Claim 23 (Canceled).**

**Claim 24 (Currently Amended):** A computer program product according to claim 2123, wherein the instruction of the program is executed first by the first virtual machine and then by the second virtual machine, and, instead of performing the write operation, the second virtual machine checks whether the value that is to be written is present in the heap at the location that is to be written to.

**Claim 25 (Previously Presented):** A computer program product according to claim 21, wherein the program is a *Java Card Applet* intended for execution by a JCVM (*Java Card Virtual Machine*).



***Allowable Subject Matter***

7. The following is an examiner's statement of reasons for allowance:
8. The prior art of record does not teach or reasonably suggest, in the combinations and in such manner as recited in independent claims 1, 10, and 21, the data carrier has a processor which executes at least a first and a second virtual machine for each execution of the program, the program is executed both by the first and by the second virtual machine, and the first and the second virtual machine both access a common heap in a non-volatile memory of the data carrier, wherein, when an instruction of the program that contains a write operation to the common heap is being executed, a write operation is performed only by the first virtual machine, emphasis added.
9. Osen (EP 1,271,317 A1, art of record) does not teach multiple virtual machines. Osen further does not teach, nor does it suggest, two virtual machines where a write operation is performed only by the first virtual machine. Furthermore, although Osen teaches comparing the contents of a common heap between the first and second execution of a program, Osen does not teach that the "operating state" of the program is compared.
10. A contemporary review article regarding attacks on smart cards, (Rankil 2003, art of record), teaches a smart card repeating a calculation more than once and comparing the result (similar to Osen). However, Rankil does not teach the multiple virtual machines, particularly comparing "operating state" and replacing each write operation with a check (compare) operation.
11. Any comments considered necessary by applicant must be submitted no later than the payment of the issue fee and, to avoid processing delays, should preferably accompany the issue fee. Such submissions should be clearly labeled "Comments on Statement of Reasons for Allowance."

***Cited Prior Art***

12. Sinha et al. (US 7,228,426 B2) is cited as a method for detecting software malfunction by multiple execution of the same program, in a security context. This method may be distinguished from the present claims in that it lacks a virtual machine, lacks multiple virtual machines, lacks replacing the write operation in a virtual machine, and lacks comparing program execution during operation of the program.
13. "Oblivious Hashing: A Stealthy Software Integrity Verification Primitive" (Chen et al, 2003) is cited as a method for detecting software malfunction by multiple execution of the same program, in a security context. This method may be distinguished from the present claims in that it lacks a virtual machine, lacks multiple virtual machines, lacks replacing the write operation in a virtual machine, and lacks comparing operating state during execution of the program.

***Conclusion***

14. Any inquiry of a general nature or relating to the status of this application or concerning this communication or earlier communications from the Examiner should be directed to **Erika Kretzmer** whose telephone number is **(571) 270-5554**. The Examiner can normally be reached Monday through Thursday, 9:30am-6:00pm Eastern Time. If attempts to reach the examiner are unsuccessful, the Examiner's supervisor, **Tuan Dam** can be reached at **(571) 272-3695**.
15. Information regarding the status of an application may be obtained from the Patent Application Information Retrieval (PAIR) system. Status information for published applications may be obtained from either Private PAIR or Public PAIR. Status information for unpublished applications is available through Private PAIR only. For more information about the PAIR system, see <http://portal.uspto.gov/external/portal/pair> . Please direct questions on access to the Private PAIR system to the Electronic Business Center (EBC) at **866.217.9197** (toll-free).
16. Any response to this action should be mailed to:

Art Unit: 2192

**Commissioner of Patents and Trademarks**

**Washington, D.C. 20231**

or faxed to 571-273-8300. Hand delivered responses should be brought to the United States

Patent and Trademark Office Customer Service Window:

Randolph Building

401 Dulany Street

Alexandria, VA 22314.

/Erika Kretzmer/  
Examiner, Art Unit 2192  
April 26, 2010

/Tuan Q. Dam/  
Supervisory Patent Examiner, Art Unit 2192